Sistem Pelaporan Infrastruktur Dinas Bina Marga dan PSDA Kota Salatiga Menggunakan *NodeJs* Berbasis Web

Christine Dewi¹⁾, Anjar Widhyo Sasongko^{*2)}

^{1,2)}Fakultas Teknologi Informasi, Program Studi Teknik Informatika Universitas Kristen Satya Wacana Salatiga

Dinas Bina Marga dan PSDA kota Salatiga memiliki tugas pelayanan masyarakat dalam hal menyediakan infrastruktur kebinamargaan dan sumber daya air di kota Salatiga. Dinas Bina Marga dan PSDA kota Salatiga saat ini belum memiliki sistem informasi untuk melaporkan kerusakan infrastruktur di kota Salatiga. Pelaporan kerusakan infrastruktur saat ini dilakukan melalui pemberitahuan langsung ke kantor dinas, pemberitahuan melalui telepon, pemberitahuan melalui *email*, atau pemberitahuan melalui sms. Dari informasi tersebut petugas masih diharuskan untuk melakukan *survey* ke lokasi laporan kerusakan. Pada penelitian ini dilakukan pengembangan sistem pelaporan infrastruktur berbasis web menggunakan *nodeJs*. Pengembangan sistem dilakukan dengan metode *prototyping*. Hasil dari penelitian ini dapat membantu pihak dinas Bina Marga dan PSDA kota Salatiga dalam menyajikan laporan masalah pada infrastruktur. Detail informasi laporan berupa gambar dan letak lokasi masalah dapat menjadi bahan pertimbangan pihak dinas dalam mengambil langkah selanjutnya.

Keywords: Pelaporan Infrastruktur, *NodeJs*, Bina Marga dan PSDA Kota Salatiga.

I. PENDAHULUAN

Teknologi informasi dan komunikasi merupakan bentuk dari perkembangan teknologi saat ini. Peran teknologi informasi dan komunikasi sangat besar, tidak hanya menjadi sarana penunjang melainkan sebagai sarana utama dalam bekerja. Pemanfaatan teknologi informasi dan komunikasi dapat digunakan sebagai alat bantu pendukung pengambilan keputusan. Pemanfaatan teknologi informasi dan komunikasi juga terjadi di instansi pemerintahan. Pemerintah pusat juga mendorong agar setiap instansi pemerintah meningkatkan pemanfaatan teknologi informasi dan komunikasi. [1]

Dinas Bina Marga dan Pengelolaan Sumber Daya Air (PSDA) kota Salatiga merupakan salah satu bagian dari satuan kerja petugas daerah (SKPD) pemerintah kota Salatiga. Dinas Bina Marga dan PSDA memiliki tugas pelayanan masyarakat dalam hal menyediakan infrastruktur kebinamargaan dan sumber daya air di kota Salatiga. Demi meningkatkan mutu pelayanan terhadap masyarakat kota Salatiga dan sekitarnya, Dinas Bina Marga dan PSDA telah melakukan berbagai upaya. Proses penyediaan infrastruktur tidak hanya berhenti pada tahap pembangunan, melainkan juga tahap perawatan dan pemeliharaan. Dalam menangani kasus kerusakan infrastruktur, pihak dinas memiliki skala prioritas yang nanti akan menentukan dalam pengambilan tindakan perbaikan. Pihak dinas telah membentuk tim cepat tanggap yang dapat melakukan perbaikan secara langsung terhadap kerusakan dengan skala kecil. Namun ruang cakup kota Salatiga yang cukup luas yaitu $\pm 57,36 \text{ Km}^2$ dengan panjang ruas jalan kota $\pm 133,400 \text{ km}$, total jembatan ± 68

jembatan dan terbatasnya jumlah petugas di lapangan serta minimnya informasi mengenai kerusakan menjadi kendala dalam menanggulangi kerusakan. [2]

Dinas Bina Marga dan PSDA telah menyediakan berbagai fasilitas bagi masyarakat untuk memberikan informasi mengenai infrastruktur yang ada pada kota Salatiga. Informasi dapat diberikan melalui pemberitahuan langsung ke kantor dinas, pemberitahuan melalui telepon, pemberitahuan melalui *email*, pemberitahuan melalui sms. Pihak dinas pernah meluncurkan program yang diharapkan dapat menyaring informasi dari masyarakat melalui *SMS gateway*, namun informasi yang diperoleh belum cukup untuk mewakili kondisi sebenarnya karena informasi yang didapat berupa teks. Dari informasi tersebut petugas masih diharuskan untuk melakukan *survey* ke lokasi laporan kerusakan.

Perumusan masalah dalam penelitian ini bagaimana merancang dan membuat aplikasi yang dapat mengumpulkan dan mengolah data pelaporan infrastruktur serta memantau persebaran lokasi masalah pada infrastruktur yang menjadi tanggung jawab dinas Bina Marga dan PSDA kota Salatiga. Fokus pada penelitian ini yaitu merancang sistem berbasis web serta perancangan *database* pada sistem. Tujuan dari penelitian ini yaitu membuat sistem yang mempermudah pekerjaan petugas dinas serta menyelesaikan masalah pengumpulan dan pengolahan data persebaran masalah infrastruktur di dinas Bina Marga dan PSDA kota Salatiga.

II. TINJAUAN PUSTAKA

Penelitian dengan judul “Sistem Informasi Geografis Pelaporan Masyarakat (SIGMA) Berbasis Foto *Geotag*”, membahas mengenai laporan masyarakat kepada pihak pemerintah menggunakan media foto digital yang telah ditambahkan informasi didalamnya yang bertujuan agar pihak pemerintah dapat memantau lokasi permasalahan infrastruktur dalam masyarakat seperti kerusakan jalan dan jembatan, penerangan jalan yang tidak berfungsi, tempat pembuangan dan pembakaran sampah ilegal, pembakaran lahan, toilet tidak sehat dan pemetaan rumah tidak layak huni. [3]

Pada penelitian kedua dengan judul “Pengimplementasian *Dashboard* Berbasis Web Gis Sebagai *Tools Monitoring* Reklame (Studi Kasus Kota Surabaya)”, dibahas mengenai aplikasi untuk membantu dalam pengawasan persebaran papan reklame di kota surabaya yang disajikan dalam sebuah peta. [4]

Pada penelitian selanjutnya dengan judul “Pemanfaatan *Google Maps API* Untuk Mencari Lokasi SPBU Terdekat Di Kota Jepara Dan Kudus Dengan Teknologi *Node-Js*”, membahas tentang pemetaan SPBU pada Kota Jepara dan Kudus dengan memanfaatkan teknologi *NodeJs* dan *Google Maps API*. [5]

Perbedaan antara penelitian ini dengan penelitian sebelumnya yaitu teknologi yang digunakan serta studi kasus pada penelitian. Penelitian ini membahas tentang permasalahan yang terjadi di dinas Bina Marga dan PSDA kota Salatiga mengenai pengolahan data informasi laporan kerusakan infrastruktur.

Pembuatan aplikasi ini dibagi menjadi dua tahap yaitu pembuatan *backend* aplikasi dan *frontend* aplikasi. Pada pembuatan *backend* aplikasi digunakan teknologi *NodeJs* sebagai *web server*, *expressjs* sebagai kerangka kerja (*framework*) pada *NodeJs*, *MySQL* sebagai *database*. Sedangkan pada *frontend* aplikasi digunakan *AngularJs* sebagai pengolah data serta *bootstrap* sebagai pengolah tampilan aplikasi.

MySQL adalah sebuah perangkat lunak sistem manajemen basis data *SQL* (*database management system*) atau *DBMS* yang *multithread*, *multi-user*. Digunakan dalam menyimpan dan mengolah basis data pada aplikasi komputer. [6]

Framework atau *software framework* merupakan suatu kerangka kerja yang dapat digunakan dalam mengembangkan aplikasi. Dalam *Framework* berisi kumpulan fungsi umum yang telah disusun untuk memenuhi tugas dasar yang sering digunakan oleh pengembang aplikasi. Dengan adanya *framework* proses pengembangan aplikasi menjadi lebih cepat dan mudah.

NodeJs adalah sebuah *platform* baru yang menarik yang dikembangkan oleh Ryan Dahl, memungkinkan pengembang bahasa pemrograman *JavaScript* untuk membuat *server* dengan kinerja yang sangat tinggi dengan memanfaatkan mesin *V8 JavaScript Google*, dan *asynchronous I/O*. [7]

ExpressJs merupakan sebuah *platform* aplikasi *web NodeJs* yang minimal dan fleksibel, menyediakan banyak

fitur yang lengkap untuk membangun aplikasi *web* tunggal, *multipage* dan *hybrid*. [8]

AngularJs adalah salah satu *framework MVC* dalam membangun aplikasi *web* berbasis *Javascript*, merupakan teknologi yang sangat populer dan memiliki keunggulan dalam dokumentasi yang terstruktur dan lengkap. [9]

Bootstrap adalah kerangka kerja yang ditujukan untuk meningkatkan proses pengembang *web frontend* lebih cepat dan lebih mudah. Kerangka kerja ini dibuat untuk semua orang dengan tingkat keahlian berbeda, untuk semua perangkat dengan berbagai bentuk dan untuk semua *web* dengan berbagai ukuran. [10]

Google Maps API merupakan pustaka *Javascript* yang disediakan oleh perusahaan *Google* untuk memberikan citra satelit, peta *streetview* dan peta interaktif.

III. METODE PENELITIAN

Penelitian ini diselesaikan melalui beberapa tahapan penelitian. Adapun tahapan-tahapan penelitian yaitu: (1) Analisa kebutuhan dan pengumpulan data, (2) Perancangan Sistem, (3) Perancangan aplikasi, (4) Implementasi dan pengujian, (5) Penulisan laporan hasil penelitian.

Gambar 1. Tahapan Penelitian

Pada Gambar 1 dapat dijelaskan bahwa pada tahap pertama dilakukan analisa kebutuhan serta melakukan pengumpulan data. Pada tahap ini dilakukan dengan melakukan observasi dan wawancara secara langsung terhadap dinas Bina Marga dan PSDA kota Salatiga mengenai proses bisnis yang berlaku serta kendala yang terjadi selama ini. Tahap selanjutnya yaitu Perancangan Sistem yang meliputi perancangan UML (*Unified Modelling Language*) seperti perancangan *use case diagram*, *sequence diagram*, *class diagram*. Selain merancang UML dilakukan perancangan *database* yang terdiri dari tabel-tabel untuk menyimpan data pada aplikasi. Pada tahapan selanjutnya yaitu perancangan aplikasi/ program dilakukan proses pembuatan program berdasarkan kebutuhan yang telah ditetapkan dalam tahapan sebelumnya. Selanjutnya tahapan ke-empat yaitu implementasi dan pengujian sistem serta

analisis hasil pengujian. Pada tahap ini peneliti melakukan implementasi aplikasi yang telah dibuat serta melakukan pengujian. Tahapan terakhir yaitu dimana setiap hasil penelitian dalam tahapan diatas mulai dari awal hingga selesai dilaporkan dalam bentuk tulisan.

Gambar 2 Metode *Prototype*

Dalam perancangan aplikasi digunakan metode perancangan *Prototype*. Metode *Prototype* merupakan sebuah metode dalam pengembangan *software* yang bertahap dan berulang. Pada Gambar 2 dapat dijelaskan terdapat tahapan dalam metode *prototype* yaitu *Listen to customer*, *Build/revise mockup*, dan *Customer test drives mockup*.

Tahap pertama *Listen to customer* yaitu mengumpulkan kebutuhan dari aplikasi dengan mendengarkan keluhan pihak Bina Marga dan PSDA kota Salatiga. Pada tahap ini didapatkan informasi kebutuhan aplikasi yaitu bagaimana mengatasi masalah pengumpulan dan pengolahan data masalah pada infrastruktur yang ada. Selain dari itu juga dibutuhkan aplikasi yang dapat memantau persebaran masalah yang terdapat pada infrastruktur. Tahap selanjutnya *Build/revise mockup* yaitu dilakukan perancangan dan pembuatan *prototype system* sesuai dengan kebutuhan pelanggan. Kemudian tahap berikutnya *Customer test drives mockup* yaitu uji coba oleh pelanggan/pengguna yang kemudian dilakukan evaluasi. Evaluasi dilakukan dengan mengumpulkan kekurangan yang ada pada program/aplikasi sesuai dengan kebutuhan pelanggan. Pengembang aplikasi kemudian mendengarkan kembali masukan dari pelanggan/pengguna guna memperbaiki *prototype*.

Gambar 3 Usecase Diagram

Pada Gambar 3 menunjukkan *use case diagram sistem* yang digunakan pada perancangan sistem. Terdapat 2(dua) buah aktor yaitu *Admin* dan *Public*, aktor *admin* merupakan pegawai dinas Bina Marga dan PSDA kota Salatiga yang bertugas mengatur data pada sistem. *Admin* mengatur data

user, data *report* dan data *category*. Ketiga *use case* merupakan *input* untuk sistem pada aktor *admin*. Pada *usecase* pengaturan data *report* terdapat *include use case Print Report Data* yang merupakan *output* dari sistem. Aktor kedua adalah *Public*, merupakan pengguna aplikasi yaitu pegawai dinas atau masyarakat umum. Aktor *public* dapat mengatur data *user* dan data *report*. Pada kedua aktor diwajibkan untuk melakukan *login* untuk mendapatkan hak akses seperti yang telah dijelaskan. Jika tidak melakukan *login* kedua aktor tersebut hanya memiliki akses untuk melihat data *report*.

Gambar 4 Activity Diagram Mengirim Report

Pada Gambar 4 merupakan *activity diagram mengirim report* yang menggambarkan proses transaksi yang dilakukan saat mengirim laporan. Aktivitas dimulai dari *user* melakukan *login*, yang kemudian dilakukan proses validasi oleh sistem. Apabila data *login* tidak *valid*, maka sistem akan mengembalikan ke tahap *login*. Bila *user* telah memasukan data yang *valid* maka sistem akan menampilkan menu utama. *User* diharuskan mengisi *form report* yang telah disediakan. Kemudian *user* mengirim *report* lalu sistem menyimpan *report* tersebut. Setelah berhasil menyimpan, sistem akan menampilkan data terbaru yang ada dalam sistem.

Gambar 5 Activity Diagram Print Report

Pada Gambar 5 dapat dijelaskan aktivitas yang hanya dapat dilakukan oleh *user* dengan hak akses *admin*. Sistem menyediakan sebuah layanan untuk mencetak data *report* kedalam format dokumen. Untuk melakukan hal tersebut, *user* diharuskan untuk melakukan *login*. Jika proses *login* berhasil, maka sistem akan menampilkan menu utama.

Kemudian *user* memilih data pada menu *report*. Setelah data yang akan di cetak telah siap, *user* memberi perintah cetak dengan menekan tombol *print*. Setelah sistem menerima perintah *print*, maka sistem akan mengkonversi data tersebut ke dalam bentuk dokumen *pdf* atau dokumen *excel*.

Gambar 6. Class Diagram Aplikasi Pelaporan

Gambar 6 dapat dijelaskan penggambaran dari tiap *class* yang terdapat pada aplikasi yang akan dikembangkan. Berdasarkan Gambar 6 terdapat 2 jenis *class* yang digunakan yaitu *class model* dan *class route*. Untuk melakukan komunikasi antara aplikasi dengan *database* digunakan *class model* sedangkan pengolahan data dari model dan menampilkan data pada *user interface* digunakan *class route*. Aplikasi ini terdiri dari 3 *class model* yaitu : *ModelUsers*, *ModelReports* dan *ModelCategories*. *Class router* yang terdapat pada aplikasi ini yaitu : *routeUsers*, *routeReports* dan *routeCategories*. Tiap *class* yang terdapat pada aplikasi dirangkai sesuai dengan metode *Model Route View*. *Class view* pada aplikasi ini hanya berisi kode *html* saja sehingga tidak dimasukkan dalam *class diagram*.

IV. HASIL DAN PEMBAHASAN

Hasil dari penelitian ini yaitu sebuah aplikasi yang dapat membantu *user* untuk melaporkan dan memantau masalah yang terjadi pada infrastruktur. Aplikasi memiliki 2 hak akses *user* yang berbeda yaitu *user* dengan hak akses *admin* dan *user* dengan hak akses publik. Setiap *user* yang akan menggunakan aplikasi tidak diwajibkan melakukan proses *login*. Namun *user* yang tidak melakukan proses *login* ke dalam sistem tidak dapat mengolah data sehingga *user* hanya dapat melihat data. *User* yang melakukan proses *login* ke dalam sistem dengan hak akses *admin* dapat melakukan olah data laporan milik semua *user* seperti pada Gambar 7.

Gambar 7. Halaman daftar laporan semua *user*

Gambar 7 merupakan halaman yang menampilkan data laporan milik semua *user*. Halaman daftar laporan digunakan untuk melihat seluruh laporan yang ada pada sistem. Proses penyajian data ke dalam bentuk tabel interaktif menggunakan *datatables javascript library*. Penggunaan *datatables javascript library* memiliki keunggulan yaitu interaksi *user* dengan data menjadi lebih interaktif. Data laporan berasal dari laporan yang dilakukan oleh semua *user* dengan menggunakan aplikasi. Sedangkan *user* yang melakukan proses *login* dengan hak akses publik hanya dapat melihat dan mengolah laporan milik *user* tersebut.

Gambar 8. Halaman daftar laporan pada peta

Laporan yang dikirimkan mengandung informasi mengenai lokasi laporan yang berupa data *latitude* dan *longitude*. Informasi mengenai lokasi laporan kemudian diolah oleh sistem dan menghasilkan tampilan berupa *marker* pada peta seperti Gambar 8. *User* dapat memantau persebaran masalah dengan melihat *marker* yang terdapat pada peta.

Kode Program 1 Fungsi menampilkan peta dan *marker*

```
1. <ng-map default-style="true" zoom="11"
   center="-7.3259873,110.4985348">
2.   <info-window id="myInfoWindow">
3.     <div ng-non-bindable style="color:
       blue">
4.       <h4>{{s.category}}</h4>
5.       <p>{{s.name}} :
       {{s.description}}</p>
6.     </div>
7.   </info-window>
8. <ng-map default-style="true" zoom="11"
   center="-7.3259873,110.4985348">
9.   <info-window id="myInfoWindow">
10.    <div ng-non-bindable style="color:
       blue">
11.      <h4>{{s.category}}</h4>
12.      <p>{{s.name}} : {{s.description}}</p>
13.    </div>
14.   </info-window>
15. <marker position="-
    7.3259873,110.4985348" centered="true"
    title="Hello" on-
    dragend="getpos($event)"
    animation="Animation.BOUNCE"
    draggable="true"></marker>
16. <marker ng-repeat="c in reports"
    position="{{c.latitude}},{{c.longitude}}"
    title="{{c.name}}"
    id="{{c.id_report}}" on-
    click="showCity(event, c)"></marker>
17. </ng-map>
```


Kode program 1 merupakan potongan kode untuk menampilkan peta yang memanfaatkan layanan dari *Google Maps Javascript API*. Pada baris 1 merupakan kode yang digunakan untuk menentukan lokasi peta serta perbesaran peta saat pertama kali ditampilkan. Tiap marker yang ditampilkan jika di *click* akan menampilkan *info-window* yang berisi keterangan laporan, fungsi untuk menampilkan *info-window* terdapat pada baris 2 hingga baris 7. Pada kode 9 merupakan potongan kode yang digunakan untuk menampilkan *marker* pada peta. Untuk menampilkan seluruh *marker* digunakan fungsi perulangan dan data *binding* pada *angularJs*. Dengan menggunakan fitur tersebut mempermudah dalam penyajian data.

Gambar 9. Halaman tambah data laporan

User yang telah melakukan proses *login* dapat mengirim laporan dengan mengisi data pada *form* pelaporan seperti pada Gambar 9. Pengirim laporan diwajibkan untuk mengisi semua *field*, mulai dari memilih *file* gambar, memilih *category*, mengisi deskripsi tentang laporan dan mengisi lokasi laporan. Untuk mengisi lokasi, *user* harus memindahkan salah satu *marker* yang terdapat di peta pada Gambar 8. *Marker* yang dapat dipindah hanya ada 1 yaitu *marker* yang bergerak ke arah atas dan bawah. Setelah semua *field* pada *form* selesai di isi *user* menekan tombol *submit*.

Kode Program 2 Fungsi input data laporan

```
1.   var fs           = require('fs');
2.   app.post('/api/addReport/',
upload.any(), function(req, res, next) {
3.       session = req.session;
4.       //upload file
5.       if(req.files){
6.           req.files.forEach(function(f
7.           le){
8.               var fileName = (new
Date).valueOf()+"-"+file.originalname;
9.               var idReport = "r"+(new
Date).valueOf()
10.              var idUser = req.session.uniqueID;
11.
12.              fs.rename(file.path,
'public/images/'+fileName, function(err){
13.                  if (err)throw err;
14.              var report = {
15.                  id_report : idReport,
16.                  id_user : idUser,
17.                  picture : fileName,
18.                  description :
req.body.description,
```

```
19.                  id_category :
req.body.category,
20.                  latitude :
req.body.latitude,
21.                  longitude :
req.body.longitude,
22.              };
23.              todo.create(report, res);
24.          });
25.      });
26.  }
27.  });
```

Kode program 2 merupakan potongan kode untuk menambah data laporan. Kode program 2 pada baris 1 merupakan *module* pada *NodeJs* yang berfungsi untuk menangani pengelolaan data berupa *file*. Kode program 2 pada baris 2 merupakan fungsi untuk menangani proses *request response* dari *http method*. Kode program 2 baris 8 hingga 10 merupakan deklarasi variabel yang meliputi variabel *fileName* dan *idReport* yang memanfaatkan fungsi dari *NodeJs* untuk mengambil data tanggal, dan variabel *idUser* yang memanfaatkan fungsi *session* pada *NodeJs*. Selanjutnya *file* gambar disimpan pada *folder* yang telah disiapkan menggunakan fungsi olah data *file* yang disediakan oleh *NodeJs* dapat dilihat pada kode program 2 baris 12. Pada kode program 2 baris 14 hingga 21 dapat dijelaskan deklarasi variabel *report* yang memiliki berisi kumpulan data-data yang akan disimpan pada sistem. Data disimpan ke dalam sistem dengan menjalankan fungsi *create* seperti pada kode program 2 baris 23.

Gambar 10. Halaman galeri laporan

Gambar 10 dapat dijelaskan laporan juga tersaji dalam bentuk galeri, *user* dapat melihat semua gambar laporan yang tersimpan pada sistem. Informasi yang ditampilkan tidak hanya berupa gambar, namun *user* dapat melihat informasi mengenai deskripsi gambar dengan menempatkan *cursor* di depan gambar yang diinginkan.

Kode Program 3 Fungsi pagination galeri

```
1.   <div id="galery" class="col-
lg-4 col-sm-6" ng-repeat="c in reports |
startFrom:currentPage*pageSize |
limitTo:pageSize" style="background-color:
#333333 ;">
2.       .....
3.       $scope.currentPage = 0;
```

```

4.         $scope.pageSize = 6;
5.         $scope.numberOfPages=function() {
6.             return Math.ceil
7.             ($scope.reports.length/$scope.pageSize);
8.         }
9.         a.filter('startFrom',
function() {
10.            return function(input,
start) {
11.                start = +start;
12.                //parse to int
13.                return
input.slice(start);
14.            }
});

```

Pada kode program 3 merupakan fungsi *pagination* yang disediakan oleh *AngularJs*. Kumpulan gambar yang akan ditampilkan dimasukan dalam elemen *html* menggunakan fungsi perulangan pada *AngularJs* yaitu *ng-repeat* seperti pada kode program 3 baris 1. Selanjutnya dilakukan pembagian konten dalam beberapa *page* seperti pada kode baris 9.

Gambar 11 Tombol visibilitas dan file export

Gambar 11 merupakan tampilan tombol yang terdapat pada tabel laporan. Tampilan tombol akan muncul ketika *user* dengan hak akses *admin* melakukan *login*. Tombol *column visibillity* berfungsi untuk menyembunyikan kolom pada tabel. Tombol *copy* digunakan untuk menyalin data pada table, tombol *pdf* digunakan untuk mengeksport ke dalam bentuk *file pdf*, tombol *excel* digunakan untuk mengeksport ke dalam bentuk *file excel*, kemudian tombol *print* digunakan untuk mencetak data pada tabel secara langsung. Tombol-tombol tersebut merupakan fitur yang disediakan oleh *library datatable*.

Gambar 12. Hasil export dengan format dokumen pdf

Pada Gambar 12 dapat dijelaskan hasil dari *export* data pada tabel ke dalam bentuk *file* dokumen dengan format *pdf*. Sedangkan hasil *export* data pada tabel ke dalam bentuk *file* dokumen dengan format *excel* dijelaskan pada Gambar 13.

Gambar 13. Hasil export dengan format dokumen excel

Kode Program 4 Fungsi export file dokumen pdf

```

1.     i.ext.buttons.pdfFlash =
f.extend({}, t, {
2.         className: "buttons-pdf buttons-
flash",
3.         text: function(a) {
4.             return a.i18n("buttons.pdf", "PDF")
5.         }, action: function(a, b, d, c) {
6.             var a = c._flash, e = b.buttons.
7.             exportData(c.exportOptions),
8.             g = b.table().node().offsetWidth,
9.             i.ext.buttons.pdfFlash =
f.extend({}, t, {
10.                className: "buttons-pdf buttons-
flash",
11.                text: function(a) {
12.                    return a.i18n("buttons.pdf", "PDF")
13.                }, action: function(a, b, d, c) {
14.                    var a = c._flash, e = b.buttons.
15.                    exportData(c.exportOptions),
16.                    g = b.table().node().offsetWidth,
17.                    f =
b.columns(c.columns).indexes().map(function(a) {
18.                        return
19.                        b.column(a).header().offsetWidth / g });
20.                    a.setAction("pdf");
21.                    a.setFileName(r(c));
22.                    s(a, JSON.stringify({
23.                        title: r(c, !1),
24.                        message: "function" ==
25.                        typeof c.message ? c.message(b, d, c) :
26.                        c.message,
27.                        colWidth: f.toArray(),
28.                        orientation: c.orientation, size:
29.                        c.pageSize, header: c.header ? e.header :
30.                        null, footer: c.footer ? e.footer : null,
body: e.body
31.                    })))
32.                },
33.                extension: ".pdf",
34.                orientation: "portrait",
35.                pageSize: "A4",
36.                message: "",
37.                newline: "\n"
38.            });

```

Pada kode program 4 merupakan potongan kode yang digunakan untuk mengkonversi data pada tabel ke dalam bentuk *file* dokumen *pdf*. Pada kode program 4 baris 1 hingga

4 merupakan kode yang digunakan untuk memanggil fungsi konversi dokumen *pdf*. Sebelum melakukan proses konversi, program mendeklarasikan variabel seperti kode program 4 baris 5 hingga 8. Data yang diperoleh kemudian diubah dari data dengan tipe data JSON ke dalam bentuk data dengan tipe data *string*. Pengaturan halaman dilakukan pada kode program 4 baris 18 hingga 22.

Kode Program 5 Fungsi *export file* dokumen *excel*

```

1.      i.ext.buttons.excelFlash =
f.extend({}, t, {
2.      className: "buttons-excel buttons-
flash",
3.      text: function(a) {
4.          return
a.i18n("buttons.excel", "Excel") },
5.          action: function(a, b, d, c)
{
6.              var a = c._flash, e = 0,
7.              g =
f.parseXML(p["xl/worksheets/sheet1.xml"])
,
8.              h =
g.getElementsByTagName("sheetData")[0],
9.              d = { _rels: { ".rels":
f.parseXML(p["_rels/.rels"])}}, xl: {
_rels: { "workbook.xml.rels":
f.parseXML(p["xl/_rels/workbook.xml.rels"
]) }, "workbook.xml":
f.parseXML(p["xl/workbook.xml"])}},
"styles.xml":
f.parseXML(p["xl/styles.xml"])}},
10.          worksheets: { "sheet1.xml": g } },
"[Content_Types].xml":
f.parseXML(p["[Content_Types].xml"])}}, b
= b.buttons.exportData(c.exportOptions),
11.          .....
12.      f("sheets sheet",
d.xl["workbook.xml"]).attr("name", A(c));
13.      c.customizeData &&
c.customizeData(b);
14.      c.header && (i(b.header, e), f("row
c", g).attr("s", "2"));
15.      for (var m = 0, n = b.body.length;
m < n; m++) i(b.body[m], e);
16.      c.footer && b.footer &&
(i(b.footer, e), f("row:last c",
g).attr("s", "2"));
17.      i = l(g, "cols");
18.      f("worksheet", g).prepend(i); m = 0;
19.      for (n = b.header.length; m < n;
m++) i.appendChild(l(g, "col", { attr: {
min: m + 1, max: m + 1, width: z(b, m),
customWidth: 1 } }));
20.      c.customize &&
c.customize(d);
21.      v(d);
22.      a.setAction("excel");
23.      a.setFileName(r(c));
24.      a.setSheetData(d);
25.      s(a, "")
26.      },
27.      extension: ".xlsx"
28.      });

```

Pada kode program 5 merupakan potongan kode yang digunakan untuk mengkonversi data pada tabel ke dalam bentuk *file* dokumen *excel*. Pada kode program 5 baris 1 hingga 5 merupakan fungsi yang digunakan untuk memanggil fungsi konversi data kedalam bentuk *file excel*. Pada proses konversi dibutuhkan beberapa variabel, proses persiapan variabel ditunjukkan pada kode program 5 baris 7 hingga 10. Proses konversi data kedalam bentuk *file* dokumen dilakukan pada kode program 5 baris 12 hingga 26. Setelah data berhasil dimasukan dalam *file* dokumen, diberikan jenis *file* ekstensi seperti pada kode program baris 27.

Pengujian aplikasi dilakukan dengan menguji fungsi-fungsi pada aplikasi yang dibuat pada penelitian. Pengujian fungsi-fungsi yang terdapat pada aplikasi bertujuan untuk mengetahui apakah aplikasi yang dibangun dapat memenuhi kebutuhan pengguna. Pengujian dan hasil pengujian dapat dilihat pada tabel 1.

Tabel 1.

Tabel hasil pengujian *blackbox*

Pengujian	Kondisi	Output yang diharapkan	Output yang dihasilkan	Status pengujian
Login	Email dan password benar	Sukses login	Sukses login	Valid
	Email dan password salah maupun kosong	Gagal login	Gagal login	Valid
Menampilkan lokasi laporan pada peta	Membuka halaman laporan	Sukses tampilkan lokasi laporan	Sukses tampilkan lokasi laporan	Valid
Menambah Data Laporan	Field pada form terisi lengkap	Sukses tambah data laporan	Sukses tambahdata laporan	Valid
	Form tidak terisi lengkap	Tidak dapat menambah data laporan	Tidak dapat menambah data laporan	Valid
Ambil lokasi dari marker	Menggeser marker pada peta	Sukses mengambil lokasi	Sukses mengambil lokasi	Valid
Menampilkan gambar ke dalam galeri	Membuka halaman galeri	Sukses menampilkan gambar di dalam galeri	Sukses menampilkan gambar di dalam galeri	Valid
Menampilkan data laporan	Membuka halaman tabel laporan	Sukses menampilkan data laporan ke dalam tabel	Sukses menampilkan data laporan ke dalam tabel	Valid
Exportfile dokumen	Menekan tombol pdf	Sukses exportfilepdf	Sukses exportfilepdf	Valid
	Menekan tombol excel	Sukses exportfileexcel	Sukses exportfileexcel	Valid
Cetak data laporan	Menekan tombol print	Sukses cetak data laporan	Sukses cetak data laporan	Valid

Berdasarkan pengujian pada aplikasi dengan menggunakan metode *blackbox* dapat dijelaskan bahwa status pengujian dari tiap fungsi memiliki hasil *valid*. Dengan hasil tersebut maka dapat disimpulkan bahwa aplikasi yang dibuat sudah sesuai dengan harapan. Pengujian selanjutnya

adalah pengujian yang dilakukan oleh orang yang tidak ikut serta dalam pembuatan aplikasi atau calon pengguna aplikasi. Pengujian dilakukan terhadap masyarakat Kota Salatiga secara acak. Berdasarkan wawancara terhadap penguji aplikasi, aplikasi dapat membantu dan mempermudah dalam pengelolaan data laporan kondisi infrastruktur.

V. SIMPULAN

Berdasarkan penelitian yang telah dilakukan, maka dapat ditarik kesimpulan bahwa dengan menggunakan sistem yang dibuat dapat mempercepat serta mempermudah pekerjaan pengumpulan dan pengelolaan data laporan infrastruktur. Aplikasi ini dapat membantu pihak dinas Bina Marga dan PSDA kota Salatiga dalam menyajikan laporan masalah pada infrastruktur. Penggunaan *NodeJs* pada *server side* memudahkan dalam pembangunan aplikasi web karena mendukung komunikasi *Asynchronous* yang dapat membuat kinerja sistem menjadi lebih ringan. Selain itu penggunaan *AngularJs* pada *client side* mempermudah dalam pembangunan *Single Page Application* khususnya pada proses *data binding*.

Hasil dari pengujian menunjukkan bahwa dengan adanya sistem pelaporan dapat membantu pihak dinas Bina Marga dan PSDA kota Salatiga mengetahui detail informasi laporan. Detail informasi laporan berupa gambar dan letak lokasi masalah dapat menjadi bahan pertimbangan pihak dinas dalam mengambil langkah selanjutnya.

DAFTAR PUSTAKA

- [1] <http://pegilayanan.go.id/tentang-pegilayanan/> diakses 2 oktober 2016.
- [2] <http://www.kemendagri.go.id/pages/profil-daerah/kabupaten/id/33/name/jawa-tengah/detail/3373/kota-Salatiga> diakses 8 agustus 2016
- [3] Mardani, Ary. 2014. "Sistem Informasi Geografis Pelaporan Masyarakat (SIGMA) Berbasis Foto *Geotag*". Jurnal Sistem Informasi dan Teknologi Informasi.
- [4] Sani, Nisfu Asrul. 2015. "Pengimplementasian *Dashboard* Berbasis *Web GIS* Sebagai *Tools Monitoring* Reklame (Studi Kasus Kota Surabaya)". Seminar Nasional Sistem Informasi Indonesia.
- [5] Syafiq, Ahmad. 2016. "Pemanfaatan *Google Maps API* Untuk Pencarian Jalur Lokasi SPBU Terdekat Di Kota Jepara & Kudus Dengan Teknologi *Node-Js*". Seminar Nasional Telekomunikasi dan Informatika.
- [6] Solichin, Achmad. 2010. "MySQL 5 : Dari Pemula Hingga Mahir". Jakarta.
- [7] <http://visionmedia.github.io/masterynode/book.html> . Diakses 1 november 2016
- [8] Brown, Ethan. 2014. "Web Development with Node and Express". California : O'Reilly Media.
- [9] Seshadri, Shyam & Brad Green. 2014. "AngularJS: Up And Running". California : O'Reilly Media.
- [10] Spurlock, Jake. 2013. "Bootstrap". California : O'Reilly Media.